Neural Network Processing for Real Time, Sub-Pixel, Hyperspectral Data Extraction

Robert Stirbl, James Breckinridge, Curtis Padgett, Ayanna Howard, Tom Chrien, and Kenneth Brown

Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Drive, Pasadena, CA 91109-8099

Abstract

The goal of many applications that using hyperspectral images is to describe the constituent components of each pixel in a scene. Typically, a set of relevant end members are selected and a best fit mixing model is used to derive the proportions of these end members found in every pixel. For applications where the goal is to detect and possibly recognize targets that occupy a portion of a hyperspectral pixel, one need not *decode* the background to determine if a given target spectra is contained in the pixel signature. This takes computational resources and may introduce inaccuracies due to modeling simplifications. Our approach places each pixel in a hyperspectral scene into one of **n** classes based on its distance to a set of **n** cluster prototypes. The cluster prototypes have been previously identified using a modified clustering algorithm based on prior sensed data. Associated with each cluster is a set of linear filters specifically designed to separate á target embedded in a background signature from other typical signatures in the cluster. Each pixel is projected on this set of filters and the result is fed into a trained neural network for classification.

A detailed description of this divide and conquer algorithm is provided. We outline our methodology for generating training and testing data, describe modifications to a standard clustering algorithm incorporating a priori knowledge about the targets, explain how the linear filters are designed, and provide details in training the neural network classifier. Evaluations of the overall algorithm and each of the specified changes demonstrate the feasibility of these ideas. For pixels with embedded targets taking up no more than 10% of the area, detection rates approach 99.9% with a false positive rate of less than 10^{-4}.

1. Introduction

The objective of this project is to develop algorithms for autonomous detection and recognition of sub-pixel objects in hyperspectral data. The evaluation of these algorithms is based on inserting actual target signatures into real scenes of hyperspectral images. Each pixel in the scene is filtered through a bank of templates designed to minimize background elements and maximize potential target elements in the signature. The filtered result is then evaluated with a classifier to determine if a target is indeed present in the pixel.

The scenes used are generated by the Airborne Visible InfraRed Imaging Spectrometer (AVIRIS), an optical sensor that delivers calibrated images of the upwelling radiance in 224 spectral channels, or bands, with wavelengths from 400- 2500-nanometers (nm). AVIRIS uses a scanning mirror to sweep back and forth, producing 614 pixels for the 224 detectors during each scan. Each pixel produced by the instrument covers a range of approximately 20-meter square area on the ground, with some overlap, yielding a ground swath of about 11-km. The instrument flies aboard a NASA ER-2 airplane at approximately 20-km above sea level. AVIRIS has flown all across North America, many parts of Europe, and South America.

Imaging Spectroscopy is the acquisition of images where for each spatial resolution element in the image a spectrum of the energy arriving at the sensor is measured. These spectra are used to derive information based on the signature of the interation of matter and energy expressed in the spectrum. This spectroscopic approach has been used in the laboratory and in astronomy for many years. AVIRIS measures the upwelling radiance spectrum from 400 to 2500-nm at 10-nm resolution. From the molecular absorptions and constituent scattering characteristics expressed in this spectrum, AVIRIS is able to make quantitative measurements including detecting and identifying the surface and atmospheric constituents present in a given region.

The target data inserted into the background was developed from the Forest Radiance I experiment (August, 1995), provided by SITAC. The database contains reference signatures for a number of potentially interesting military targets. The spectra were modified to minimize the differences (e.g. number of bands sampled, quality) between the target and the AVIRIS background so as not to bias the results. The scenario we are examining involves examining ground images from AVIRIS to determine if the spectral signature from any of the 20-meter square pixels contains, as a constituent element, a member of a given (known spectral signatures) target set. The images taken represent scenery from a typical forward engagement area in a battle situation. In real time, we intend to remove atmospheric effects from the data, reduce its dimensionality using an optimal set of linear filters, and spatially locate targets in the scene with a neural network classifier. This can be accomplished on a pixel by pixel basis, allowing a massively parallel implementation. Figure 1 provides an overview of our approach for detecting a known set of objects in hyperspectral imagery. This report describes the methodology used to investigate sub-pixel detection and our initial evaluation of an algorithm suitable for highly parallel implementation specifically running on an analog, low power processor.

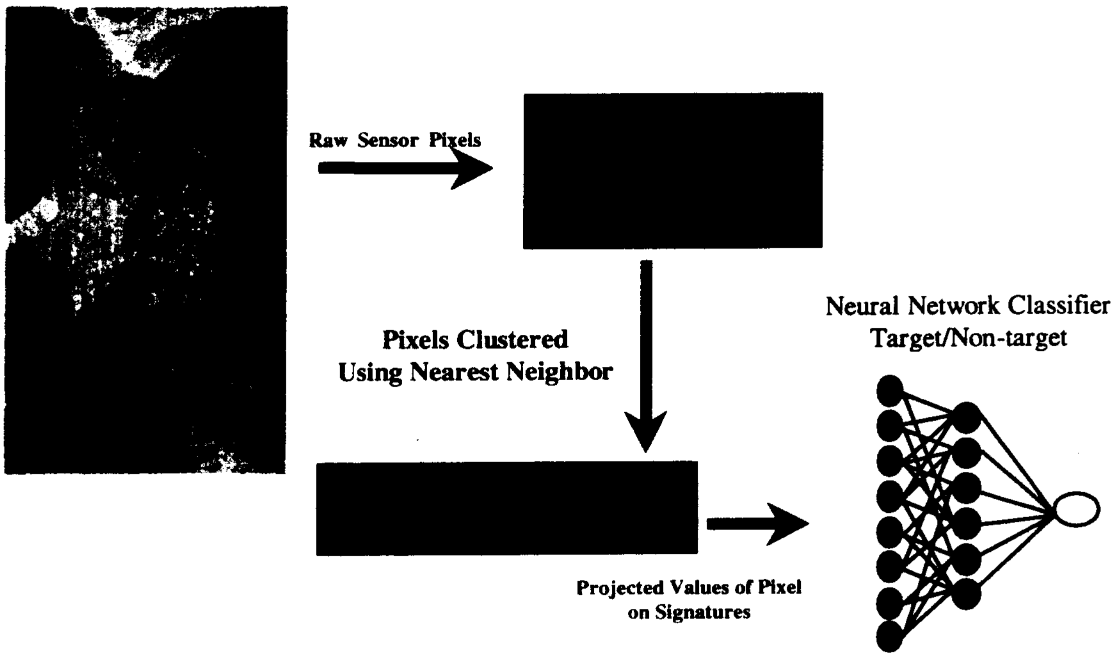
Raw Sensor Pixels
Pixels Clustered Using Nearest Neighbor
Neural Network Classifier Target/Non-target
Projected Values of Pixel on Signatures

Figure 1: The data processing path for each pixel from the sensed image. The raw pixel is corrected for atmosphere, assoicated with a particular cluster, projected onto a set of filters (eigenvectors) for that cluster and then classified with the neural network.

2. AVIRIS Data and Target Description

As described earlier, the AVIRIS sensor generates 224 bands of spectral information for each pixel. AVIRIS collects spectra sequentially, using a whiskbroom scan mechanism. The radiance from an approximately 20 m^2 patch on the ground is dispersed thru four grating spectrometers to obtain a spectrum consisting of 224 channels [7]. Each spectral channel is calibrated to units of spectral radiance by subtraction of a dark offset level, formed by an average of 64 dark measurements each scan line, and then scaled to units of radiance using a set of calibration coefficients [8]. A mild correction term, derrived by observing the signal of an onboard calibrator, is applied to correct for minor instability in the sensor's radiometric performance [9]. The spectral response function of the spectrometer channels is also measured using a scanning monochromator [8] and parameters of a best fit Gaussian function are determined.

The AVIRIS data is inverted to units of spectral reflectance using a radiative transfer model estimate of atmospheric path radiance and reflected radiance (ref 4). The model makes use of a model fit of the 940 nm water vapor feature for each AVIRIS spectrum. The presence of surface water, such as from vegetation, and reflectance slope is allowed for in the fitting process. The best fit water vapor amount is used to index a look-up table of pre-computed path and reflected radiance values. Inversion to units of reflectance is computed as the difference between the measured radiance and the LUT path radiance divided by the LUT reflected radiance. The quality of resulting reflectance spectra has been validated by field calibration experiment [8], and by qualitative inspection of the spectra.

The scenes used in this study are Cuprite (see Figure 1), and the ARM site (see Figure 5). Target spectra were obtained from ground truth measurements conducted at the FR1 experiment [11]. The target spectra are reduced to reflectance by comparing the unknown target to a known reflectance standard. The reflectance of the target is determined by computing the ratio of the signals between target and standard multiplied by the reflectance of the standard. These reflectance data were further processed using a median filter to remove noisy channels, interpolated to 1 nm resolution using a spline function, and then resampled using the appropriate AVIRIS spectral response function. These steps are required to account for the different spectral samplings of the field spectrometers and the AVIRISsensor.

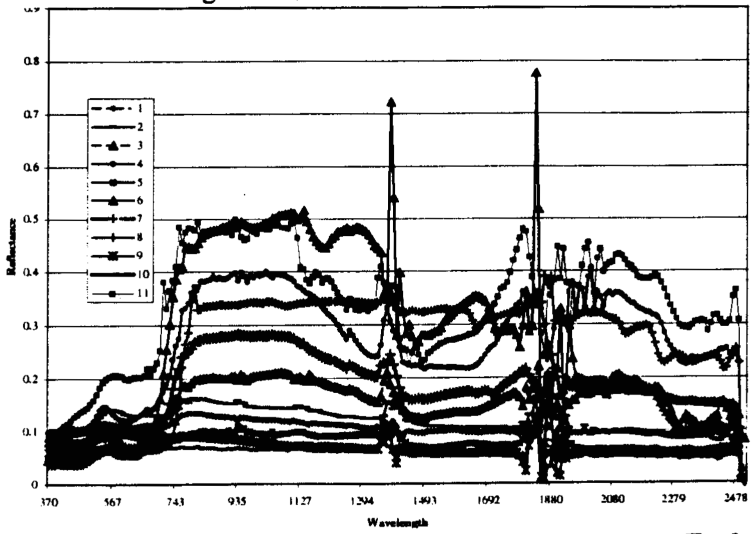

Figure 2. Average target signatures obtained from the FR1 experiment. Each example is the average of 8 signatures of a given target.

Targets are mixed into the AVIRIS scene at an arbitrary fractional level of x% by forming the composite spectra that are the sum of ***x*target + (1-x)*background***. For problems in which it made sense to examine either sensor or target noise, we defined the signal-to-noise ratio (SNR) as 50% of the reflectance value divided by the deviation of the

noise distribution. For all cases, we added randomly generated zero mean Gaussian values to inject the noise. Figure 3 provides an example of randomly drawn background pixels (from a single cluster) and pixels from the same distribution with a 10% target mix (again drawn randomly). As the caption in Figure 3 indicates, separating the mixed target examples from the background elements, in this case of a single cluster, is an extremely difficult task. The problem is made even more difficult in that the number of allowable false positives must remain quite low due to the large number of samples in a given scene. Even with a false positive rate of 1:10000, such a detection algorithm run on a scene with one million pixels would generate about 100 false alarms. The next section describes our approach for detecting the elements in Figure 3 that are targets.

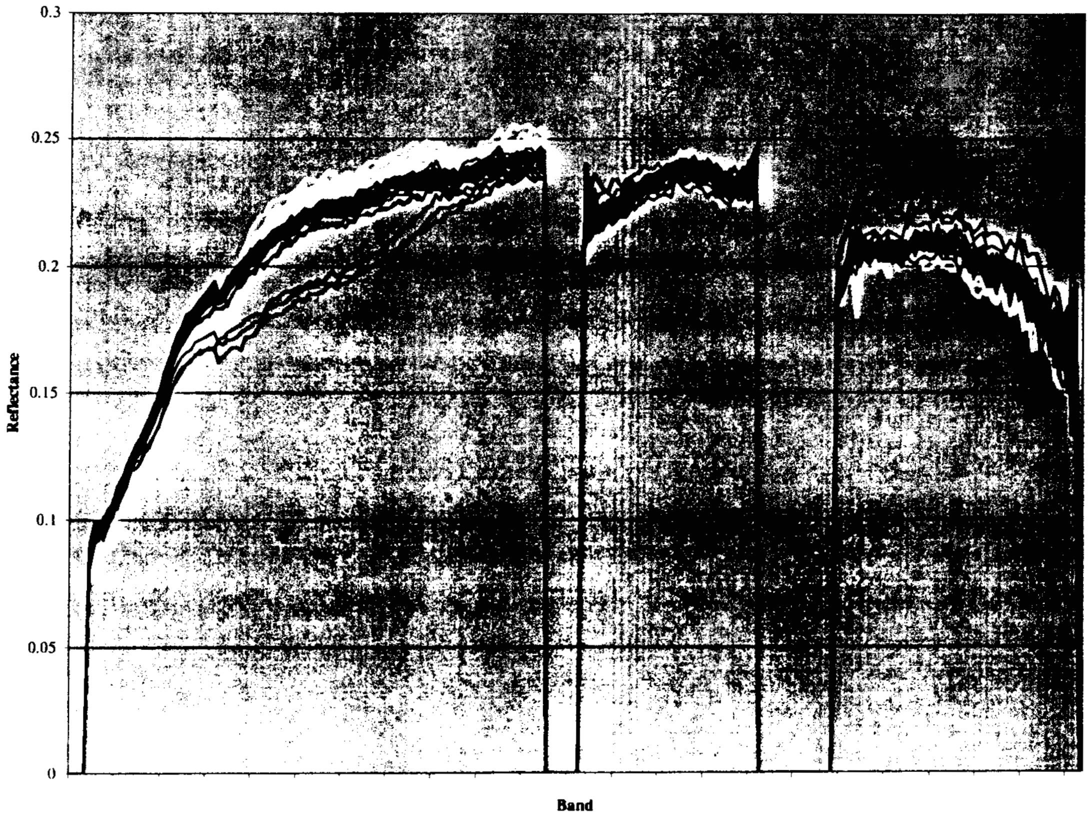

Figure 3. Randomly sampled background pixels (black) and targets mixed at 10% (white) drawn from the same cluster. The bands at 0 were not used for detection purposes (water bands). Although there seems to be considerable separation in the first third of the graph, full ½ of the target data is completely covered by the background spectra (the black lies over the white plots).

3. Algorithm Description

Given a set of targets **T**, the goal of the algorithm is to detect any element **t** $\in$ **T** in real time that is present in the hyper-spectral pixel of a scene. To meet the real time goals of the algorithm, the system developed utilizes two multi-processor boards (SHARC 16 processor cards) slotted into the main bus (PCI) of a personal computer. These 32 processors perform atmospheric correction for each sensor pixel, assign it to a cluster for evaluation, and evaluate each cluster. The evaluation is done by first projecting each pixel onto 40 orthogonal dimensions specific to each cluster and then analyzing the 32 resultant values with a previously trained neural network specific to the cluster.

The atmospheric correction routine consists of all operations on pixels necessary to remove gross atmospheric and sensor bias from the raw sensor data. This pre-processing step normalizes each pixel with respect to sun angle, water vapor content (atmospheric components of variation) and incorporates any calibration information regarding the sensor. These variants impact different portions of the spectra in stereotypical fashion and may effectively hide key features useful in the discrimination of background spectra from those containing targets. Inputs to this routine can be obtained from telemetry (for sun angle calculations), raw spectral bands (for estimating water vapor), and pre-flight (or in-flight) sensor calibration information. The goal is to achieve quality reflectance data in real time with minimal variation over key feature regions. For this study, reflectance data was used in generating all the results—we have yet to determine if standard routines are suitable for implementation in the described system (in real time).

The clustering step in the algorithm is accomplished with a simple nearest neighbor algorithm [4]. Each normalized pixel is compared to a set of pre-computed spectra prototypes. The pixel is grouped with the closest prototype (Euclidean distance). All subsequent operations on the pixel are based on the group that it is associated with. This step partitions the initial problem into **n** distributions of spectra associated with the prototype spectra. As the evaluation of the pixel is based on our a priori knowledge about these distributions, the prototypes used need to reflect the underlying distributions of the scene. How the prototypes are generated is described later in this section.

The filtering step is an orthogonal sub-space projection of each pixel. It is used to optimally (linear) separate the background pixels of each group with targets (*signal*) from those without (*noise*). Each pixel is projected onto the 32 basis vectors for its group after subtracting out the mean of the group. This is a standard technique used to reduce the dimensionality of the pixel (224->32) while preserving as much of the signal as possible. As the classifier uses the 32 resultant values to determine if the pixel contains a target, the reduced number of dimensions allows for fewer examples in learning the decision boundaries for the task (typically $O(d^2)$ for a non-linear classification problem with d dimensions).

The last step uses a neural network classifier to evaluate whether or not the pixel contains a target from **T**. The neural network for each group takes as input the projected values of the pixel and outputs a value. Values above a threshold are considered pixels with targets and those below are assigned to background. Both the threshold and neural network parameters are learned from training examples based on spectra clustered in the associated group.
The effectiveness of the evaluation requires that the prototypes generated and the spectra used in training the classifier must be derived from scenery with roughly the same distributions as encountered in the operational test. The following pseudo code outlines the important features of the algorithm.

Let p_{xy} be a raw pixel located at (x,y) in the scene:

```
for all x,y-
    (1) $p_{xy} = AC(p_{xy}, sun\_angle, sensor\_calibration)$
    (2) $p_{xy} \rightarrow min_i( \| p_{xy} - P_i \| )$
for all $p_{xy}$ in i-
    (3) if $C_i(p_{xy}^T W_i) < thr_i$ then $target(x,y)$
                          otherwise $background(x,y)$
```

Where AC ( ) is the atmospheric correction, p_{xy} is the corrected pixel, ^{T} is the transpose operation, P_i is the *closest* prototype, $\mathbf{W}_i$ is the filter set for group i, and C_i is the neural network classifier for that group. $\mathbf{thr_I}$ is threshold value discriminating between pixels with targets and those without. Obviously, in a parallel implementation, the two **for** loops can be overlapped so that the entire scene need not be segmented (2) prior to projecting the corrected pixel and evaluating it by the neural network (3).

Steps 2 and 3 in the above algorithm require prior knowledge about the distribution of pixel spectra in the scene being evaluated. This knowledge consists of:

1) A set of background prototypes derived from scene characteristics that are used to segement the scene.
2) A set of linear filters for each background type used to optimally separate targets imbedded in background pixels from other background pixel examples.
3) A set of *expert* neural network classifiers that receive as input the projections of the corrected pixels on their respective filter set. The neural networks are required to respond with 1 when targets are embedded in the pixels and –1 otherwise. The resultant output is thresholded to achieve the desired level of false positives.

The remainder of this section describes how each of these components are generated for a given set **T** of target signatures and a known distribution of background signatures in which the targets will be found. Optimizing these components directly improves the performance of the algorithm in detecting targets as will be shown in the results section.

A. Clustering

To effectively simplify the distribution of data classified by an expert neural network, we partition the incoming pixels into a number of predetermined groups by using the prototypes of a clustering algorithm (designated as the set P_i). The clustering algorithm is run on previously aquired data that reflects the distribution of the scene being analyzed. The prototypes are the averages of the clusters identified by the algorithm. To achieve good results, the clusters should partition the scene pixels into a set of **n** groups each having a narrower (and more defined) distribution of constituent pixels. This will allow us to employ filters that effectively minimze this distribution, highlighting potential pixels containing elements from the target set.

We employed two different clustering algorithms in analyzing the overall algorithm. The first algorithm is a standard clustering technique outlined in Duda and Hart [Duda & Hart]. It uses a standard least squares criterion to minimize the distance between each of **n** randomly selected groups. The criterion minimized by the clustering algorithm is:

$$(1)\ \text{cost} = \Sigma_i \Sigma_j \| p_j - P_i \|$$

where i is one of **n** clusters and p_j is a pixel in that cluster. The clustering algorithm iterates through each pixel and determines if moving the pixel to another group reduces the overall cost. If it does, the pixel is moved to the other group and the associated averages of each prototype cluster are recalculated. This continues until the moving of pixels no longer reduces the overall cost. The resultant cluster prototypes are then employed by our algorithm to segment the scene.

Two things are worth noting about this technique. Firstly, as **n** increases, the overall cost is likely to go down as a larger number of groups allow the clustering algorithm to better fit the given distribution. The narrower distributions are more amenable to masking by the filters in step **3** of our algorithm, which should result in better detection rates. Obviously, signal to noise issues, the underlying true pixel distributions, and the additional computational cost in step **2** of our algorithm effectively limit the size of **n** and make it problem dependent. Nonetheless, more groups should increase the performance of the detection algorithm. Secondly, the clustering algorithm, as described, is independent of the detection problem. It does not take into account any information that we might have concerning the target set. The size of the distribution for clusters that closely resemble targets could be quite wide thus making it more difficult to detect target pixels in them. As targets are typically camouflaged to resemble the background, this is a potentially serious problem. Similarly, it makes little sense to have narrow distributions for clusters whose elements highly contrast with the targets (making them easy to detect).

What is needed are narrow distributions of backgrounds that resemble targets in our set and wider distributions for those backgrounds that contrast with them. To accomplish this, we modified the criterion given in (1) to reflect target knowledge and then evaluate this modified algorithm in the results section. The change in (1) consists of simply weighting each pixel by a term reflecting its closeness with elements in the target set. The modified criterion is given by:

$$(2)\ \ \text{cost} = \Sigma_i\, w_j \Sigma_j \,\|\, p_j - P_i \,\|$$

where w_j is:

$$(3)\ \ w_j = 1/\Sigma_s \,\|\, p_j - t_s \,\|$$

were t_s is an element of $\mathbf{T}$, the target set. Pixels that are close to targets will be weighted more in the cost of the clustering algorithm than those further away allowing the clustering algorithm to naturally provide more resources (groups) to those background types.

B. Filter Sets

The filters associated with a given prototype are derived from the distribution of its background pixels (essentially ***noise*** with respect to detection) and the distribution of potential targets (at some mix percentage) embedded in that background (the ***signal***). This can be optimally separated to maximize the *signal to noise* ratio between the two groups using directed principal components analysis (DPCA). The filters generated by DPCA can be identified off line by determing the target spectra associated with each cluster using an appropriate mixing model. To characterize the distribution for cluster i the covariance matrix, $\mathbf{R}_i$, is found for pixels in the group (without targets). We can also characterize the mixed target-pixel distribution instances associated with cluster i by its covariance matrix, $\mathbf{S}_i$.

We are interested in finding a set of orthogonal basis vectors (filters) W_i, that maximizes the expected signal to noise ratio of these two distributions defined by their respective pixel sets. The generalized eignevector solution to the following equation accomplishes this:

$$(4)\ \ \mathbf{S}_i\, \mathbf{W}_i = \lambda\, \mathbf{R}_i\, W_i$$

The set of filters defined by W_i is the directed components used in step (**3**) of our algorithm. They essentially steer the eigenvector solution away from dimensions of high noise variance in a linearly optimal fashion. If the noise characterized by $\mathbf{R}_i$ is Gaussian, equation (4) results in the standard solution found by principal component analysis. Others have employed similar techniques in sub-pixel detection problems with slightly different formulations and have reported good results [1,3].

C. Classification

Step **3** in our algorithm involves classifying each pixel's projection on W_i with a neural network. The networks are trained with data drawn from the two distributions used to determine W_i, $\mathbf{R}_i$ and $\mathbf{S}_i$. The expert network for class *i* is required to respond with 1 for elements drawn from $\mathbf{S}_i$ and -1 from those drawn from $\mathbf{R}_i$. We used a simple feed forward model employing 10 sigmoidal hidden units trained with back propogation to get the desired result. The output can then be thresholded to achieve the desired detection rate or false positive rate by examining the receiver operator curves.

While the formulation of the problem studied here is linear—the targets were linearly mixed into the backgrounds, the actual classification problem resulting from multiple target signatures, the distributions resulting from the clustering, and the resultant projections of the corrected pixels, need not be. In addition, certain non-linear elements are often present in real problems—sensor noise, pixel correction, or actual mixing of target and background that lead us to believe that a non-linear classifier is more suitable for this problem. Evidence that this is the case is provided later in the results section.

4. Results

We evaluated the overall performance of the algorithm using the described target set and two AVIRIS scenes (Cuperite copper mine in New Mexico and Midwestern farmland). We were interested in examining its performance as the size of the target relative to the background pixel varied, as target measurement and sensor noise changed, and in the influence that different backgrounds had on the detection rate. In addition, we explored how the algorithmic choices affected the detection rate. In particular we were interested in determining whether or not performance gains were achieved with:

1. The use of DPCA instead of the more traditional PCA approach.
2. Incorporating target knowledge into the clustering algorithm.
3. Using a non-linear neural network over a linear classifier.

The following subsection describes our methodology used in evaluating the algorithm, the results of our evaluation on the overall algorithm performance and the analysis of the modifications we've made to the existing techniques (1-3).

A. Methodology

To provide an accurate estimate of how well the algorithm performs, we used random techniques for sampling both the training data (for clustering, covariance matrixes in DPCA, and neural network learning) and the test data for evaluating the algorithm and generating the receiver operator curves. The two scenes consisted of over ¾ of a million pixels of which less than 10% were used in developing a set of training data. Testing data consisted of randomly drawn pixels from the remaining scene. Target pixels were generated by randomly selecting spectra from the target set and linearly mixing them with arbitrary background pixels. The training data was then evaluated with either clustering technique to realize the prototypes (**P**) used in step **1** of the algorithm. Figure 5 shows the prototypes generated by a sample clustering and Figure 6 shows the ARM scene segmented with those prototypes. A sub-sample of the training data (1000 examples each) was used to generate the covariance matrixes $\mathbf{R}_i$ and $\mathbf{S}_i$. The generalized eigenvector solutions *W*, to these matrixes were then solved using a Matlab routine

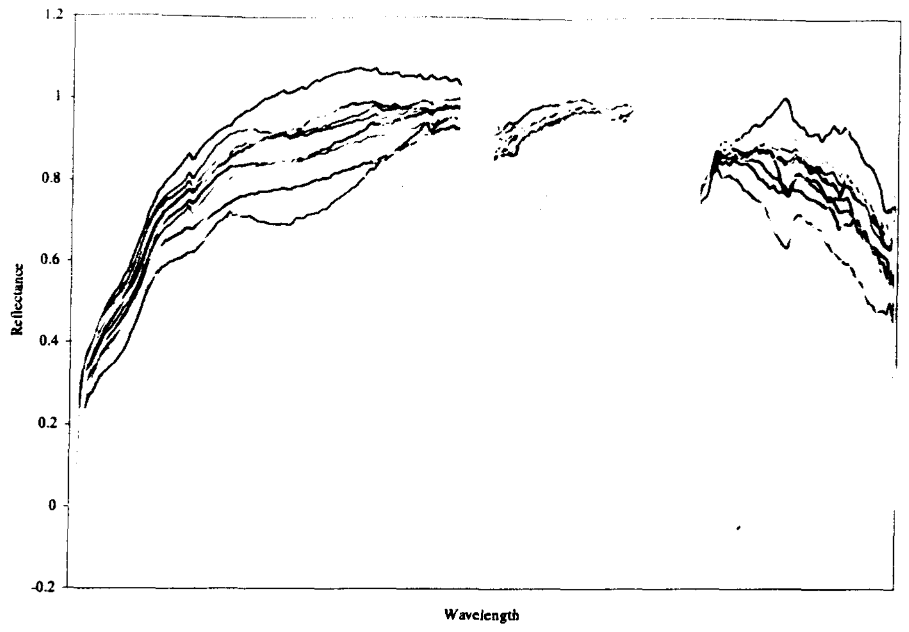

Figure 4. Example prototypes obtained from the clustering algorithm. The prototypes in this case were generated from the Cuperite mining scene with 16 clusters.

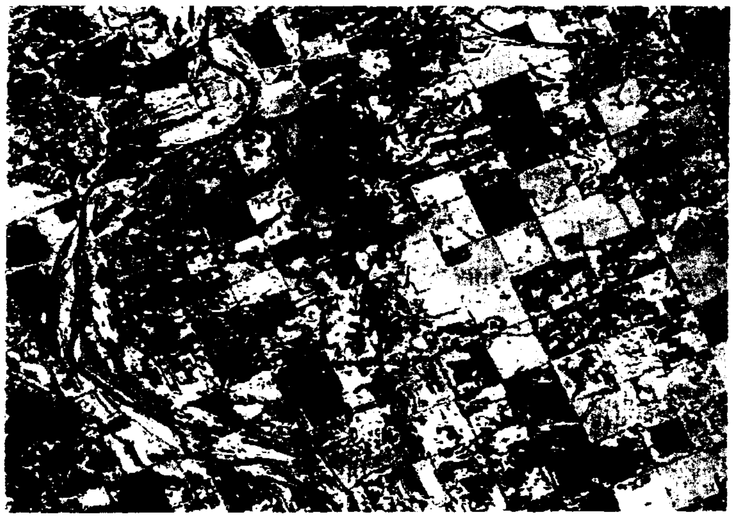

Figure 5. The ARM scene segmented with the prototypes. This process corresponds to step 2 in the algorithm.

Training data for the neural network was again drawn from the set of training pixels. In addition, a portion of the training data for the network was used to halt training (a hold out set) as described in Haykin [5]. Training of the networks used 5000 examples, ½ target and ½ background pixels. The hold out set consisted of 2500 examples not trained upon. It is used to stop training in order to prevent over learning on the training data which tends to decrease generalization. As we have no prior knowledge as to what features in W the network would find useful, we modified it by multiplying each dimension i, in W, by $1/\sigma_i$, where σ is the standard deviation of example pixels projected on that eigenvector. This is important as there are typically large variations in the projected values on the different eigenvectors (due to the way they were calculated). Important features in lower order principal components could easily be considered noise by the network unless this step is taken. The modified vectors in *W* generate the W used in step 3 of our algorithm.

B. Experiment 1: Detection as target size varies

One of the major goals in this project was to determine how large the target had to be in relation to the pixel. For search or characterization scenarios, larger pixels for a given size target represents more ground coverage and allows for a wider search to be conducted in the same amount of time and for the sensor to be on a higher (and presumably safer) flying platform. Of course, all sub-pixel detection algorithms will benefit by larger targets relative to pixel size (among other variants) making it difficult to draw comparisons between algorithms. Our tests are designed to provide some indication of just what target/pixel ratio makes sense to describe what algorithmic modifications make sense, and finally to describe our methodology from which we could analyze alternate algorithms.

In the first experiment, we examined four mix ratios of target and background. The algorithm was evaluated with 16 clusters derived using the unmodified clustering algorithm. No noise was added to the targets or the background pixels. The natural (sensed) variation in the example spectra contained in a cluster or the target gives an upper bound on the performance of the algorithm for the given parameters (it is unlikely that adding noise will result in better performance). Obviously changes in the parameters will move the detection rates and these will be investigated in later experiments. Figure 6 plots the receiver operator curves (detection vs. false positives) for mix percentages of 2%, 5%, and 10%. 25% target mix resulted in 100% detection (not shown) with no false positives. The plots shown consist of averaging the results from 8 of the 16 clusters (over 50,000 pixels sampled).

From the graph, it is clear that even substantial improvement to a 2% target mix is unlikely to result in a reasonable detection algorithm (unless used as a pre-processor). The detection rate is simply too low at any useful level of false positives. Even the 5% graph shown here is unacceptable with respect to false positives. However it is significantly better than 2% and is low enough to allow us to evaluate algorithmic modifications. The 10% graph is simply too good—any algorithmic changes would require hundreds of thousands of pixel evaluations to show significance (in improvement) while it should be readily apparent at the 5% level. For purely pragmatic reasons our evaluations of algorithmic parameters and modifications are based on target mixes of 5%.

C. Experiment 2: Changing the number of clusters (n) and the type of clustering

Segmenting the scene into **n** clusters is the factor that has the largest impact on detection rates. Increasing the size of **n** means that the filter set and classifier associated with a particular cluster will, on average, have as input a more compact distribution. Similarly, making more responsive cluster--the distributions of those clusters of low contrast with the target set narrower—means that each classifier gets about the same problem difficulty. This makes discrimination between target and background considerably easier and reduces variance in detection rates across clusters.

Figure 7 shows the receiver operator curves for clustering with **n** equal to 16, 64, or 128. The size of the neural networks, the type of clustering, target mixtures, etc., are held constant for each cluster size. There is some natural variation in the clustering algorithm (initial clusters are randomly chosen and local minima exist) so that some variation is to be expected in the overall detection rates. However the differences between the detection rates for the various size clusters are significant. Figure 7 clearly shows the improvement obtained by increasing the number of clusters.

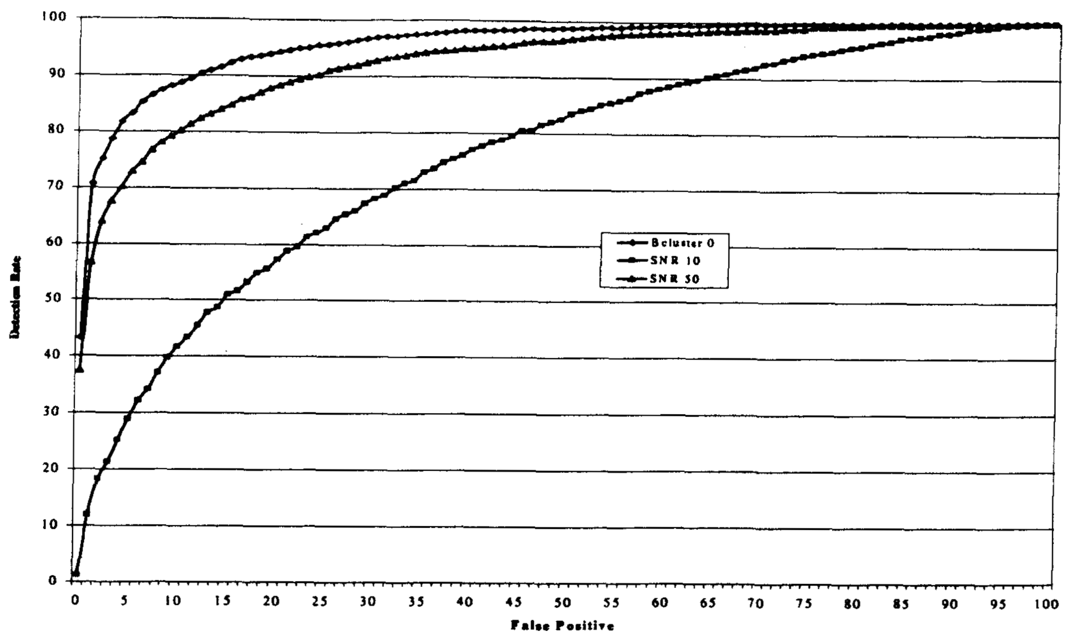

Figure 6. Receiver operator curves for neural network output as target mix in background pixel varies as a percentage of pixel size. The three curves show 10%, 5%, and 2% target mix. At 25% mix (not shown) the algorithm had 100% detection with no false positives.

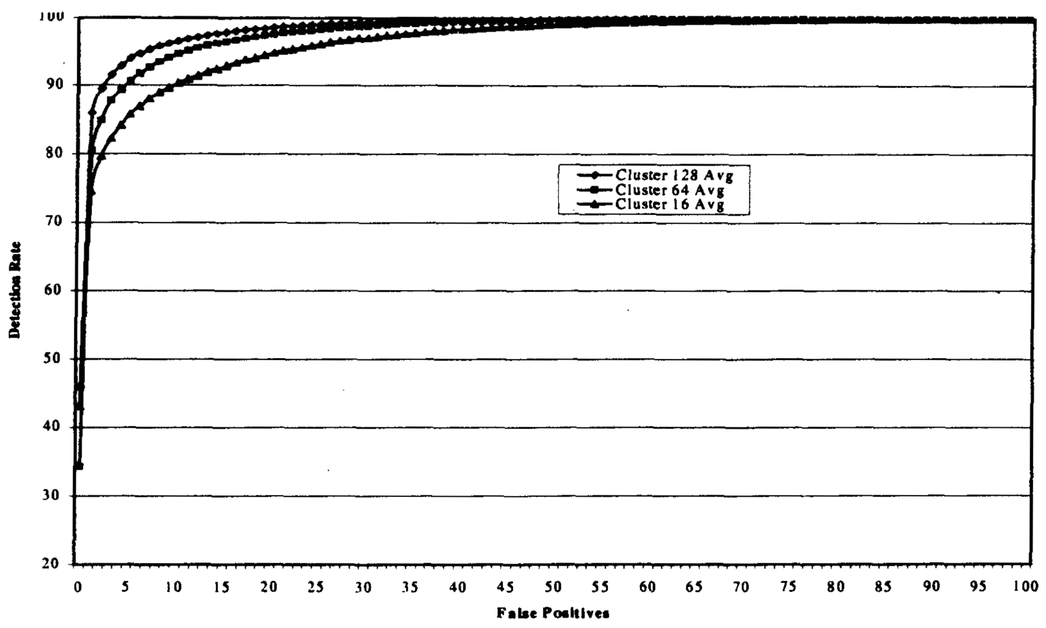

Figure 7. Receiver operator curves for neural network output as number of clusters varies. Each curve is averaged over the same number of pixels to realize the detection rate for the given level of false positives. Considerable improvement is achieved by increasing the number of clusters.

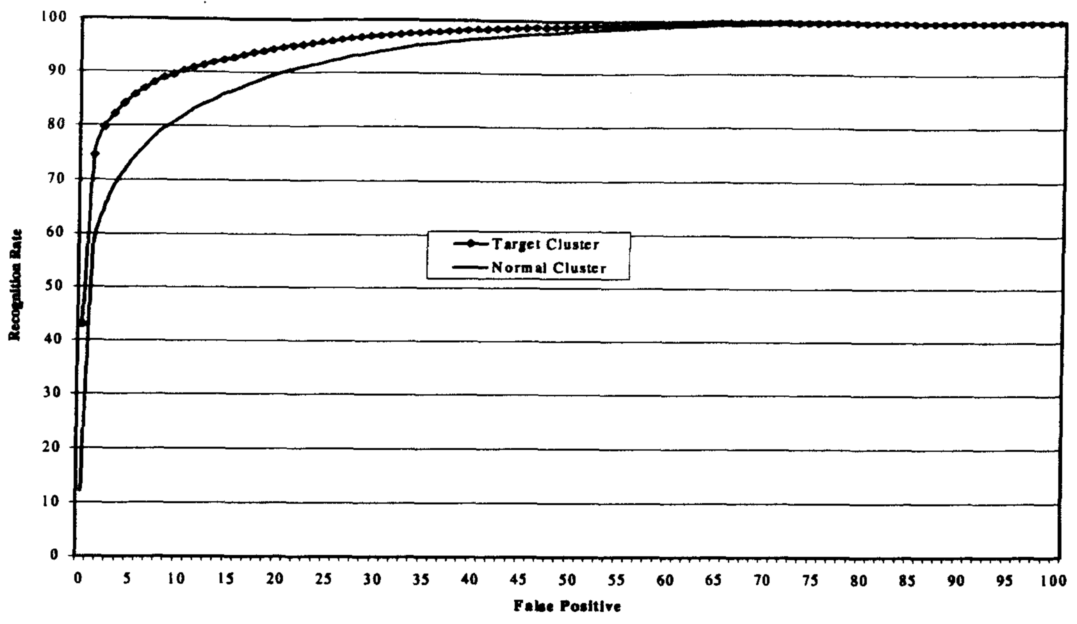

Figure 8. Receiver operator curves for neural network output for the two different clustering methods. The lower curve is the average detection rate for the original clustering algorithm (16 clusters). The modified algorithm (16 clusters) with target knowledge clearly outperforms the original clustering.

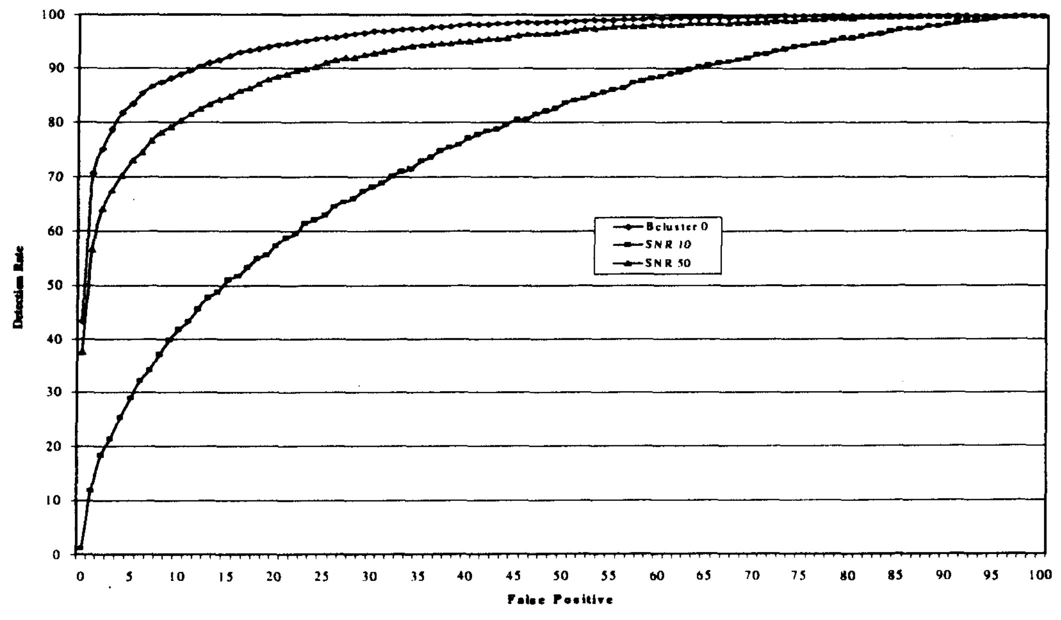

Figure 9. Receiver operator curves for neural network output as noise is added to pixel elements. Effectively, this type of noise measures algorithmic

performance as sensor quality changes. Shown are the curves for the original, 10, 50 SNR

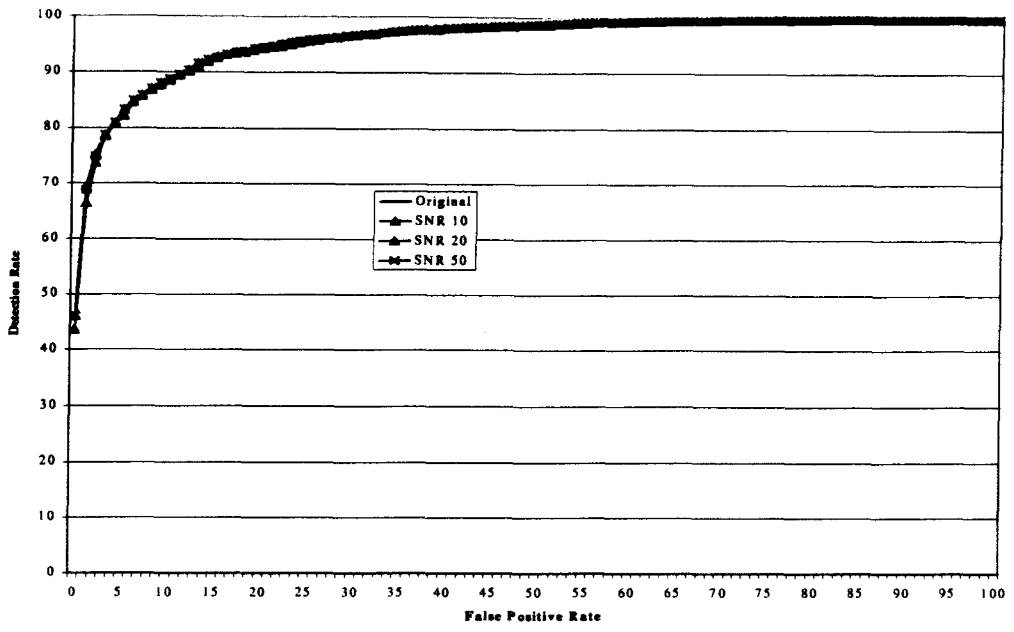

Figure 10. Receiver operator curves for neural network output as SNR varies on the mixed target elements. The targets shown include the original target set along with targets modified using SNRs of 10, 20, and 50.

Adding target knowledge to the clustering algorithm demonstrates the impact background contrast has on the detectability of targets mixed into that background. Just as increasing the number of clusters makes the discrimination task easier for each expert classifier, making each cluster have about the same intrinsic difficulties also improves the average detection rate for the scene. Figure 8 shows the improvement obtained by adding target knowledge to the clustering process.

D. Experiment 3: Adding Noise to the Targets and Sensor

The filter sets are specifically designed to minimize the impact of noise in the detection process. However, significant noise is bound to impact the detection rates. The next two figures examine the impact of noise on the target set (how well we know the target spectra) and the sensor (how well the sensor images the scene). Figure 9 gives the receiver operator curves for the original cluster and with SNR of 50 and 10. In this case, Gaussian noise was added to the pixel prior to evaluation after step 1 in the algorithm. This simulates poor calibration, noise in calculating reflectance, or simple sensor noise. The impact on the algorithm is substantial at the 5% mix percentage shown. A much smaller impact on performance is seen at the 10% target mix for the levels of noise injected here (possibly indicating that the 5% mix is quite near the detectability limit).

Figure 10 provides the impact of noise degradation on the target set. This represents less than perfect knowledge about the target spectra or its properties. Again, Gaussian noise is added (per band) to targets at 10, 20, and 50 SNR and the resultant receiver operator curves

5. Conclusion

A novel detection algorithm and our evaluation methodology are described here. The detection algorithm was shown to perform detection at a rate of over 99% with false positives less than 0.1% on a set of targets mixed at 10% with background pixels. For larger targets, the detection rates approach 100% (at 25% mix, the algorithm was perfect).

We also show a number of modifications to the basic detection approach. We used clustering to divide up the problem into a number of partitions so that each cluster can be analyzed independently. The use of directed principal components instead of the standard principal component analysis also provided a significant performance boost. A modified clustering algorithm was shown to more equitably (with respect to the expert networks) divide up the problem, resulting in performance improvements. Finally, our analysis shows that improvements (or control) of variants prior to segmenting the scene may impact the detection in more fundamental ways than does knowledge about the target data.

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

[1] Harsanyi, J. and Chang, C. "Hyperspectral Image Classification and Dimensionality Reduction: An Orthogonal Subspace Projection Approach", *IEEE Transactions on Geoscience and Remote Sensing*, vol. 32, pp. 779-785, July 1994.

[2] Kim, B. and Landgrebe, D. "Hierarchical Classifier Design in High-Dimensional, Numerous Class Cases", *IEEE Transactions on Geoscience and Remote Sensing*, vol. 29, pp. 518-528, July 1991.

[3] Devaux, M., Bertrand, D., and Qannari, M. "Application of Principal Component Analysis of NIR Spectral Collection after Elimination of Interference by a Least-Squares Procedure", *Applied Spectroscopy*, vol. 42, pp. 1020-1023, 1988.

[4] Duda, H. and Hart, M. *Pattern Classification and Scene Analysis.* John Wiley & Sons. New York. 1973.

[5] Haykin, S. *Neural Networks: A Comprehensive Foundation.* Macmillan College Publishing Company. New York. 1994.

[6] Diamantaras, K. and Kung, S. *Principal Component Neural Networks: Theory and Applications.* John Wiley & Sons. New York. 1996.

[7] Vane, G., Green, R.O., Chrien, T., et al. "The Airborne Visible/Infrared Imaging Spectrometer (AVIRIS)", *Remote Sens. Environ.* Vol. 44, pp. 127-143, 1993.

[8] Chrien, T., Green, R.O., and Eastwood, M.L. "Accuracy of the Spectral and Radiometric Laboratory Calibration of the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS)", *SPIE Imaging Spectroscopy of the of the Terrestrial Environment*, vol. 1298, 1990.

[9] Green, R. "Use of Data from the AVIRIS Onboard Calibrator", *Summaries of the Fourth Annual JPL Airborne Geoscience Workshop, October 25-29,* JPL Publication 93-226, vol. 1, pp 65-68, 1993.

[10] Green, R.O., Conel, J.E., and Roberts, D.A. "Estimation of Aerosol Optical Depth, Pressure Elevation, Water Vapor and Calculation of Apparent Surface Reflectance from Radiance Measured by the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) Using A Radiative Transfer Code" *SPIE Imaging Spectrometry of the Terrestrial Environment,* vol. 1937, pp 2-11, 1993.

[11] Ground Truth Protocol, April 30, 1997, HYMSMO Program Office, Spectral Information Technology Applications Center (SITAC), 11781 Lee Jackson Memorial Highway, Suite 500, Fairfax, VA 22033, HYMSMO POC (703-591-8546).